UNITED STATES PATENT AND TRADEMARK OFFICE

UNITED STATES DEPARTMENT OF COMMERCE
United States Patent and Trademark Office
Address: COMMISSIONER FOR PATENTS
P.O. Box 1450
Alexandria, Virginia 22313-1450
www.uspto.gov

APPLICATION NO.	FILING DATE	FIRST NAMED INVENTOR	ATTORNEY DOCKET NO.	CONFIRMATION NO.
10/654,743	09/03/2003	Shmuel Hen	P16176	5235

46915 7590 04/11/2007
KONRAD RAYNES & VICTOR, LLP.
ATTN: INT77
315 SOUTH BEVERLY DRIVE, SUITE 210
BEVERLY HILLS, CA 90212

EXAMINER	
VERDI, KIMBLEANN C	
ART UNIT	PAPER NUMBER
2109	

SHORTENED STATUTORY PERIOD OF RESPONSE	MAIL DATE	DELIVERY MODE
3 MONTHS	04/11/2007	PAPER

Please find below and/or attached an Office communication concerning this application or proceeding.

If NO period for reply is specified above, the maximum statutory period will apply and will expire 6 MONTHS from the mailing date of this communication.

Office Action Summary	Application No.	Applicant(s)	
	10/654,743	HEN ET AL.	
	Examiner	Art Unit	
	Kacy Verdi	2109	

-- The MAILING DATE of this communication appears on the cover sheet with the correspondence address --

Period for Reply

A SHORTENED STATUTORY PERIOD FOR REPLY IS SET TO EXPIRE 3 MONTH(S) OR THIRTY (30) DAYS, WHICHEVER IS LONGER, FROM THE MAILING DATE OF THIS COMMUNICATION.

- Extensions of time may be available under the provisions of 37 CFR 1.136(a). In no event, however, may a reply be timely filed after SIX (6) MONTHS from the mailing date of this communication.
- If NO period for reply is specified above, the maximum statutory period will apply and will expire SIX (6) MONTHS from the mailing date of this communication.
- Failure to reply within the set or extended period for reply will, by statute, cause the application to become ABANDONED (35 U.S.C. § 133). Any reply received by the Office later than three months after the mailing date of this communication, even if timely filed, may reduce any earned patent term adjustment. See 37 CFR 1.704(b).

Status

1)☒ Responsive to communication(s) filed on *03 September 2003*.

2a)☐ This action is **FINAL**. 2b)☒ This action is non-final.

3)☐ Since this application is in condition for allowance except for formal matters, prosecution as to the merits is closed in accordance with the practice under *Ex parte Quayle*, 1935 C.D. 11, 453 O.G. 213.

Disposition of Claims

4)☒ Claim(s) *1-39* is/are pending in the application.

4a) Of the above claim(s) _____ is/are withdrawn from consideration.

5)☐ Claim(s) _____ is/are allowed.

6)☒ Claim(s) *1-39* is/are rejected.

7)☐ Claim(s) _____ is/are objected to.

8)☐ Claim(s) _____ are subject to restriction and/or election requirement.

Application Papers

9)☒ The specification is objected to by the Examiner.

10)☒ The drawing(s) filed on *July 28, 2006* is/are: a)☒ accepted or b)☐ objected to by the Examiner.

Applicant may not request that any objection to the drawing(s) be held in abeyance. See 37 CFR 1.85(a).

Replacement drawing sheet(s) including the correction is required if the drawing(s) is objected to. See 37 CFR 1.121(d).

11)☐ The oath or declaration is objected to by the Examiner. Note the attached Office Action or form PTO-152.

Priority under 35 U.S.C. § 119

12)☐ Acknowledgment is made of a claim for foreign priority under 35 U.S.C. § 119(a)-(d) or (f).

a)☐ All b)☐ Some * c)☐ None of:

1.☐ Certified copies of the priority documents have been received.

2.☐ Certified copies of the priority documents have been received in Application No. _____.

3.☐ Copies of the certified copies of the priority documents have been received in this National Stage application from the International Bureau (PCT Rule 17.2(a)).

* See the attached detailed Office action for a list of the certified copies not received.

Attachment(s)

1) ☒ Notice of References Cited (PTO-892)
2) ☐ Notice of Draftsperson's Patent Drawing Review (PTO-948)
3) ☒ Information Disclosure Statement(s) (PTO/SB/08) Paper No(s)/Mail Date *March 30, 2004*.
4) ☐ Interview Summary (PTO-413) Paper No(s)/Mail Date. _____.
5) ☐ Notice of Informal Patent Application
6) ☐ Other: _____.

DETAILED ACTION

This office action is in response to the Application filed on September 3, 2003. Claims 1-39 are pending in the current application.

Specification

1. The abstract of the disclosure is objected to because the period is missing at the end of the last sentence. Correction is required. See MPEP § 608.01(b).

Claim Rejections - 35 USC § 102

2. The following is a quotation of the appropriate paragraphs of 35 U.S.C. 102 that form the basis for the rejections under this section made in this Office action:

> A person shall be entitled to a patent unless –
>
> (e) the invention was described in (1) an application for patent, published under section 122(b), by another filed in the United States before the invention by the applicant for patent or (2) a patent granted on an application for patent by another filed in the United States before the invention by the applicant for patent, except that an international application filed under the treaty defined in section 351(a) shall have the effects for purposes of this subsection of an application filed in the United States only if the international application designated the United States and was published under Article 21(2) of such treaty in the English language.

3. Claims 1-2, 15-16, and 27- 28 are rejected under 35 U.S.C. 102(e) as being anticipated by U.S. Patent 6,754,736 B1 to Ogawa et al. (hereinafter Ogawa).

4. As to claims 1 and 27, Ogawa teaches a method and an article of manufacture for communicating with a device, comprising:

executing a kernel module in a memory (driver 20, Fig. 1 operating in Kernel 50, Fig. 2, Kernel resident in memory) ;

executing at least one kernel thread (kernel thread T0, Fig. 9) in the memory (Kernel 50, Fig. 2, Kernel resident in memory) to handle calls to device driver functions

for the kernel module (calls input/output function for the kernel device driver, col. 9, lines 16 and 66); and

executing, with the at least one kernel thread (kernel thread T0, Fig. 9), calls to device driver functions (T0 calls input/output function to activate asynchronous thread A0, col. 9, lines 66-67) for the kernel module (driver 20, Fig. 1, device driver, col. 9, line 16) running in a kernel context (T0 and A0 runs in kernel space, Fig. 9).

5. As to claim 2, Ogawa teaches the method of claim 1, wherein the kernel module spawns at least one kernel thread (kernel thread T0 activates asynchronous thread A0 (other kernel threads, col. 9, lines 22-24)) to execute the calls to the device driver functions for the kernel module (T0 calls input/output function to activate asynchronous thread A0, col. 9, lines 66-67, asynchronous threads (An), execute input/out request, 64, Fig. 10).

6. As to claim 15, Ogawa teaches a system, comprising:

a network device (network interface controller 22, Fig. 1, corresponds to one of the network adaptors A0-An, Fig. 12, col. 4, lines 46-51, each network adaptor represent one network device in a system, col. 1, lines 59-63);

a memory (main storage device consisting of rom and ram, 82, Fig. 20);

a processor executing code to perform (cpu, 81, Fig.20 executes program and data of Fig. 21):

(i) execute a network device driver (Communication Management Unit 17, Fig. 1) in memory to control the network device (directly controls a network interface controller 22, Fig. 1, col. 20, lines 4-11);

(ii) execute a kernel module in the memory (driver 20, Fig. 1 operating in Kernel 50, Fig. 2, Kernel resident in memory);

(iii) execute at least one kernel thread (kernel thread T0, Fig. 9) in the memory (Kernel 50, Fig. 2, Kernel resident in memory) to handle calls to device driver functions for the kernel module (calls input/output function for the kernel device driver, col. 9, lines 16 and 66); and

(iv) execute, with the at least one kernel thread (kernel thread T0, Fig. 9), calls to device driver functions (T0 calls input/output function to activate asynchronous thread A0, col. 9, lines 66-67) for the kernel module (driver 20, Fig. 1, device driver, col. 9, line 16) running in a kernel context (T0 and A0 runs in kernel space, Fig. 9).

7. As to claims 16 and 28, Ogawa teaches the system of claim 15 and the article of manufacture of claim 27, wherein the kernel module spawns at least one kernel thread (kernel thread T0 activates asynchronous thread A0 (other kernel threads, col. 9, lines 22-24)) to execute the called device driver functions (T0 calls input/output function to activate asynchronous thread A0, col. 9, lines 66-67, asynchronous threads (An), execute input/out request, 64, Fig. 10).

Claim Rejections - 35 USC § 103

8. The following is a quotation of 35 U.S.C. 103(a) which forms the basis for all obviousness rejections set forth in this Office action:

> (a) A patent may not be obtained though the invention is not identically disclosed or described as set forth in section 102 of this title, if the differences between the subject matter sought to be patented and the prior art are such that the subject matter as a whole would have been obvious at the time the invention was made to a person having ordinary skill in the art to which said subject matter pertains. Patentability shall not be negatived by the manner in which the invention was made.

9. Claims 3-10, 13, 17-23, 25, 29-35, and 38 rejected under 35 U.S.C. 103(a) as being unpatentable over U.S. Patent 6,754,736 B1 to Ogawa et al. (hereinafter Ogawa) in view of "Linux Device Drivers, 2nd Edition" by J. Corbet and A. Rubini (hereinafter Corbet).

10. As to claims 3, 17, and 29 Ogawa does not teach the method of claim 1, the system of claim 15, and the article of manufacture of claim 27, further comprising:

accessing, with one kernel thread, device information from the device; and

buffering the accessed device information.

However Corbet teaches Linux™ device drivers comprising:

accessing, with one kernel thread (request module create new "kernel thread" process, chapter 11, section 11.1.2., lines 4) , device information from the device (net_device structure contains information of the device, chapter 14, section 14.3 and device methods for network interface (adapter) used to obtain device information, chapter 14, section 14.3.2.2., line 1); and

buffering the accessed device information (stored in net_device structure as module in kernel , e.g. capabilities data in kernel, Fig. 2-1, linking a module to the kernel, Chapter 2, pages 1 and 2, data in structured inserted by driver for new interface into global list of network devices, chapter 14, section 14.2.1., lines 7-8).

It would have been obvious to a person of ordinary skill in the art at the time the invention was made to have modified the invention of Ogawa with the teachings of accessing and buffering device information from Corbet because these feature would have provided the kernel driver of Ogawa with a mechanism for supporting a number of

administrative tasks, such as setting addresses, modifying transmission parameters, and maintaining traffic and error statistics (Chapter 14, page 1, lines 28-29 of Corbet).

11. As to claims 4, 18 and 30, Ogawa as modified by Corbet teaches the method of claim 3, the system of claim 17, and the article of manufacture of claim 29, wherein a kernel module function requests device information (get stats function part of the device methods of kernel- driver interface, chapter 14, section 14.3.2, lines 4-5, chapter 14, section 14.3.2.2, page 2, lines 21-25 of Corbet), further comprising: in response to the request for the device information, accessing the buffered device information (get stats function accesses the net_device structure (e.g. capabilities data in kernel, Fig. 2-1) to return the net_device_stats, chapter 14, section 14.3.2.2., page 2, lines 21-25 of Corbet).

12. As to claim 5, 19, and 31 Ogawa as modified by Corbet teaches the method of claim 1, the system of claim 15, and the article of manufacture of claim 27, wherein the kernel thread accesses buffered device information periodically and independently of kernel module requests for the device information (hard_start_xmit method initiates transmission of a packet utilizing the net_device structure (e.g. capabilities data in kernel, Fig. 2-1), chapter 14, section 14.3.2.2, page 1, lines 22-26 of Corbet)

13. As to claims 6, 20, and 32 Ogawa as modified by Corbet teaches the method of claim 1, the system of claim 15, and the article of manufacture of claim 27, further comprising:

buffering a parameter list (e.g. multicast list, stored in net_device structure as module in kernel , e.g. capabilities data in kernel, Fig. 2-1, linking a module to the kernel, Chapter 2, pages 1 and 2 of Corbet); and

setting device parameters in the buffered parameter list to values provided by kernel module functions (e.g. multicast list, kernel uses the method dev->set_multicast_list to notifiy driver whenever the list of valid multicast addresses is changed, dev->set_multicast_list, chapter 14, section 14.13, page 2, lines 1-3, and section 14.13.1, lines 5-11 of Corbet).

14. As to claims 7, 21, and 33, Ogawa as modified by Corbet teaches the method of claim 6, the system of claim 20, and the article of manufacture of claim 32, further comprising: setting a flag indicating that the kernel thread needs to set parameters at the device to device parameter values set in the parameter list (whenever dev->flags is modified (eg. IFF_PROMISC) the function kernel uses the method dev->set_multicast_list is invoked to reprogram the hardware filter chapter 14, section 14.13.1, lines 8-9 of Corbet).

15. As to claim 8, Ogawa as modified by Corbet teaches the method of claim 6, further comprising: spawning (e.g. fork) a kernel thread (kmod.thread is called by kernel to access modules, chapter 11, section 11.1, lines 11-14 of Corbet) to set device parameters to parameter values buffered in the parameter list (kmod thread would use kernel- driver interface to invoke method dev->set_multicast_list to notifiy driver whenever the list of valid multicast addresses is changed, dev->set_multicast_list, and reprogram hardware filter, chapter 14, section 14.13, page 2, lines 1-3, section 14.13.1,

lines 5-11, section 14.3.2, lines 4-5, and section 14.3.2.2, page 2, lines 21-25 of Corbet).

16. As to claims 9, 22, and 34 Ogawa as modified by Corbet teaches the method of claim 7, the system of claim 21, and the article of manufacture of claim 27, wherein the kernel thread spawned (kmod.thread is called by kernel to access modules, chapter 11, section 11.1, lines 11-14 of Corbet) to set device parameter values (kmod thread would use kernel- driver interface to invoke method dev->set_multicast_list to notifiy driver whenever the list of valid multicast addresses is changed, dev->set_multicast_list, and reprogram hardware filter, chapter 14, section 14.13, page 2, lines 1-3, section 14.13.1, lines 5-11, section 14.3.2, lines 4-5, and section 14.3.2.2, page 2, lines 21-25 of Corbet) processes the parameter list to locate buffered parameter values (e.g. dev->flag modified) and set the device parameters to the buffered parameter values (e.g. kernel-driver interface to invokes method dev>set_multicast_list chapter 14, section 14.13, page 2, lines 1-3, section 14.13.1, lines 5-11, section 14.3.2, lines 4-5, and section 14.3.2.2, page 2, lines 21-25 of Corbet).

17. As to claims 10, 23, and 35 Ogawa as modified by Corbet teaches the method of claim 7, the system of claim 21, and the article of manufacture of claim 34, wherein the kernel thread processes the parameter list by further performing:

applying a lock on information in the parameter list including the located buffered parameter values (kmod thread would use kernel- driver interface to set spinlock_t xmit_lock on the sk_buff of the net_device structure, chapter 14, section 14.3.2.3., page 2, lines 12-18, chapter 14.5, packet transmission, lines 29 of Corbet);

after applying the lock, copying the parameter values from the parameter list to a temporary buffer (hard_start_transmit method called by kernel to put data packet (sk_buff) on out going queue (buffer), chapter 14, section 14.5, packet transmission, lines5-7 of Corbet), wherein the device parameters are set to the parameter values from the parameter list in the temporary buffer (copy of sk_buff is passed as parameter to net_device in snull_tx method call, chapter 14, section 14.5, packet transmission, line 29 of Corbet) ; and

releasing the lock after copying the parameter values from the parameter list to the temporary buffer (hard_start_xmit function releases spinlock (xmit_lock) on function return, chapter 14, section 14.5.1, lines 2-3 of Corbet).

18. As to claims 13, 25 and 38, Ogawa as modified by Corbet teaches the method of claim 10, the system of claim 23, and the article of manufacture of claim 35, further comprising: after releasing the lock, executing device driver functions (hard_start_xmit function releases spinlock (xmit_lock) on function return, and function can be called again, chapter 14, section 14.5.1, lines 2-3 of Corbet) to configure the device with the parameter values in the temporary buffer (copy of sk_buff is passed as parameter to net_device in snull_tx method call, chapter 14, section 14.5, packet transmission, line 29 of Corbet).

19. Claims 11, 12, 14, 24, 26, 36, 37, and 39 are rejected under 35 U.S.C. 103(a) as being unpatentable over U.S. Patent 6,754,736 B1 to Ogawa et al. (hereinafter Ogawa) in view of "Linux Device Drivers, 2nd Edition" by J. Corbet and A. Rubini (hereinafter

Corbet) as applied to claims 10, 23, and 35 above, and further in view of "Synchronization in Portable Device Drivers" by Stein J. Ryan (hereinafter Ryan).

20. As to claims 11, 24, and 36, Ogawa as modified by Corbet does not the method of claim 10, the system of claim 23, and the article of manufacture of claim 35, further comprising:

disabling higher priority contexts before locking the parameter list; and

enabling the higher priority contexts after releasing the lock on the parameter list.

However Ryan teaches disabling higher priority contexts before locking the parameter list (device driver disables first and second stage interrupt processing when using spinlock, page 21, right column, lines 10-11); and

enabling the higher priority contexts after releasing the lock on the parameter list, (releasing spinlock enables interrupt processing, and potentially executes a queued second stage handler, page 21, left column, section 4.2, lines 2-3, and right column lines 25-26).

It would have been obvious to a person of ordinary skill in the art at the time the invention was made to have further modified the invention of Ogawa as modified by Corbet with the teachings of disabling and enabling higher priority contexts when using spinlock from Ryan because these feature would have provided the spinlock of Corbet with a mechanism for enabling and disabling interrupt processing to prevent deadlocks (page 21, section 4.2, lines 2-3 of Ryan.

21. As to claims 12 and 37, Ogawa, as modified by Corbet, as modified by Ryan teaches the method of claim 11, and the article of manufacture of claim 36, wherein the

higher priority context comprises a bottom half or Interrupt Request (IRQ) context (e.g. hardware context) (first stage interrupt processing is done by manipulating the hardware interrupt mask of the interrupt controller, page 21, right column lines 1 and 29-30 of Ryan).

22. As to claims 14, 26, and 39, Ogawa, as modified by Corbet, as modified by Ryan teaches the method of claim 1, the system of claim 15, and the article of manufacture of claim 27, further comprising:

initiating, with the kernel module, an access request with respect to device information (get stats function part of the device methods of kernel- driver interface, chapter 14, section 14.3.2, lines 4-5, chapter 14, section 14.3.2.2, page 2, lines 21-25 of Corbet);

disabling any higher priority contexts capable of accessing the device information (device driver disables first and second stage interrupt processing when using spinlock, page 21, right column, lines 10-11 of Ryan);

obtaining a lock for the device information subject to the access request values (kmod thread would use kernel- driver interface to set spinlock_t xmit_lock on the sk_buff of the net_device structure, chapter 14, section 14.3.2.3., page 2, lines 12-18, chapter 14.5, packet transmission, lines 29 of Corbet);

providing the kernel module access to the device information (device methods for network interface (adapter) used to obtain device information to the kernel, chapter 14, section 14.3.2.2., line 1 of Corbet);

releasing the lock (releasing spinlock enables interrupt processing, page 21, left column, section 4.2, lines 2-3 of Ryan); and

enabling all higher priority contexts that were disabled (releasing spinlock enables interrupt processing, and potentially executes a queued second stage handler, page 21, left column, section 4.2, lines 2-3, and right column lines 25-26 of Ryan).

Conclusion

23. The prior art made of record and not relied upon is considered pertinent to applicant's disclosure.

U.S. Patent 5,136,709 to Shirakabe et al. discloses an operating system generation method of a computer, in which a symbolic name is converted into an identification code, which is further converted into an address.

U.S. Patent 5,390,301 to Scherf discloses a method of communication between a device driver and a system kernel uses a standardized generic data structure, which facilitates communication of numeric limitations as well as flags.

U.S. Patent 5,459,867 to Adams et al. discloses description tables, which can be linked to a kernel to from a device driver.

U.S. Patent 6,003,097 to Richman et al. discloses a system for configuring a network adapter of a computer without user intervention.

U.S. Patent 6,098,112 to Ishijima et al. discloses an apparatus and method for kernel level modules or drivers identifying message processing functions and for controlling execution of the message processing functions at the stream head, so that the functions are available at the stream head for adapting stream data interpretation,

stream execution behavior, and the like in accordance with various system requirements such as requirements of devices or user processes.

U.S. Patent 6,496,847 B1 to Bugnion et al. discloses a virtual machine monitor (VMM) is included in a computer system that has a protected host operating system (HOS). A virtual machine running at least one application via a virtual operating system is connected to the VMM.

U.S. Patent 6,732,138 to Browning et al. discloses a method and system are disclosed for managing access to system resources by a user process within a multitasking data processing system.

U.S. Patent 7,076,647 B2 to Roth et al. discloses a method and apparatus in which a UNIX operating system can be configured or tuned without rebooting the system.

U.S. Patent Application Publication 2004/0176942 A1 discloses method, system and program product for simulation of a network adapter for a computing unit of a computing environment. The simulation includes providing a behavioral simulation of the network adapter and mapping the behavioral simulation to system memory of the computing unit to allow for direct memory access.

Any inquiry concerning this communication or earlier communications from the examiner should be directed to Kacy Verdi whose telephone number is (571) 270-1654. The examiner can normally be reached on Monday-Friday 7:30am-5:00pm EST..

If attempts to reach the examiner by telephone are unsuccessful, the examiner's supervisor, Xiao Wu can be reached on (571) 272-7761. The fax phone number for the organization where this application or proceeding is assigned is 571-273-8300.

Information regarding the status of an application may be obtained from the Patent Application Information Retrieval (PAIR) system. Status information for published applications may be obtained from either Private PAIR or Public PAIR. Status information for unpublished applications is available through Private PAIR only. For more information about the PAIR system, see http://pair-direct.uspto.gov. Should you have questions on access to the Private PAIR system, contact the Electronic Business Center (EBC) at 866-217-9197 (toll-free). If you would like assistance from a USPTO Customer Service Representative or access to the automated information system, call 800-786-9199 (IN USA OR CANADA) or 571-272-1000.

KV
March 30, 2007

XIAO WU
SUPERVISORY PATENT EXAMINER